UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 09/802,291 | 03/08/2001 | Ashley Saulsbury | 16747015310 | 6894 |

20350 7590 01/29/2004

TOWNSEND AND TOWNSEND AND CREW, LLP
TWO EMBARCADERO CENTER
EIGHTH FLOOR
SAN FRANCISCO, CA 94111-3834

| EXAMINER | |
|---|---|
| O BRIEN, BARRY J | |
| **ART UNIT** | **PAPER NUMBER** |
| 2183 | 5 |

DATE MAILED: 01/29/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

| Office Action Summary | Application No. | Applicant(s) | |
|---|---|---|---|
| | 09/802,291 | SAULSBURY, ASHLEY | |
| | Examiner | Art Unit | |
| | Barry J. O'Brien | 2183 | |

*-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --*

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

1)☒ Responsive to communication(s) filed on *3/8/01, 8/9/01, 10/18/01*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

4)☒ Claim(s) *1-20* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-20* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

**Application Papers**

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. §§ 119 and 120**

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

13)☒ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application) since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

a) ☐ The translation of the foreign language provisional application has been received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121 since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

**Attachment(s)**

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449) Paper No(s) *4*.
4) ☐ Interview Summary (PTO-413) Paper No(s). _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other:

## DETAILED ACTION

1. Claims 1-20 have been examined.

### *Papers Submitted*

2. It is hereby acknowledged that the following papers have been received and placed on record in the file: Declaration as received on 8/9/2001, IDS as received on 10/18/2001.

### *Specification*

3. The lengthy specification has not been checked to the extent necessary to determine the presence of all possible minor errors. Applicant's cooperation is requested in correcting any errors of which applicant may become aware in the specification.

4. The applicant is requested to review the specification and update the status of all co-pending applications made mention of, replacing attorney docket numbers with current U.S. application or patent numbers when appropriate.

### *Claim Objections*

5. Claims 3, 5 and 15 are objected to because of the following informalities:

a. Regarding claim 3, the claim language recites the limitation, "wherein decode logic selects the first and second input operand registers from the plurality of general-purpose registers." Please correct the claim language to read, "wherein **said** decode logic selects the first and second input operand registers from the

plurality of general-purpose registers," so as to provide correct antecedent basis (emphasis added).

b. Regarding claim 5, the claim language recites the limitation, "and comparison logic" on lines 3-4, and again on lines 5-6, of the claim. Please correct the claim language to read, "and the comparison logic," so as to provide correct antecedent basis (emphasis added).

c. Regarding claim 15, the claim language recites the limitation, "wherein a very long instruction word comprises the compare operation." This phrase is worded awkwardly, giving a sense that there was already a very long instruction word claimed in prior claims that further comprises a compare operation. Please correct the claim language to read more similarly to, "wherein a compare operation is comprised within a very long instruction word," so that the intention of the claim is made clearer.

6. Appropriate correction is required.

### *Claim Rejections - 35 USC § 112*

7. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

8. Claim 7 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

9. Claim 7 recites the limitation "not a number value" on line 2 of the claim. There is insufficient antecedent basis for this limitation in the claim. It is unclear whether the "not a number value" in claim 7 refers to the value stored in the output register of claim 6 when the relationship between the two operands is "not a number", or whether it refers to a specific value associated with the relationship defined by "not a number" which is not claimed. Please correct the claim language to clarify the situation.

***Claim Rejections - 35 USC § 102***

10. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

11. Claims 1, 3, 5-9, 11-12, 14, 16-17 and 20 are rejected under 35 U.S.C. 102(b) as being anticipated by the *Alpha Architecture Handbook.*

12. Regarding claim 1, the *Alpha Architecture Handbook* has taught a processing core that executes a compare instruction (see p.4-113 Sec. 4.10.8), the processing core comprising:

a. A plurality of general-purpose registers comprising a first input operand register, a second input operand register and an output operand register (see p.3-2 Sec. 3.1.3),

b. A register file comprising the plurality of general-purpose registers (see p.3-2 Sec. 3.1.3),

c. Comparison logic coupled to the register file, wherein the comparison logic tests for at least two of the following relationships: less than, equal to, greater than and no valid relationship (see p.4-113 Sec. 4.10.8, CMPTLE instruction). Here, because the IEEE Floating Compare instruction is a member of the Alpha instruction set architecture, it is inherent that any processor implementing the instruction set will have logic that performs the comparisons that the instruction includes.

d. Decode logic which selects the output operand register from the plurality of general-purpose registers (see p.4-113 Sec. 4.10.8). Here, because the IEEE Floating Compare instruction includes a target register (Fc) which stores the outputted results of a comparison, it is inherent that decoding logic be present in any processor which implements the Alpha instruction set architecture so that the specified target register can be correctly located and written into.

e. A store path between the comparison logic and the selected output operand register (see p.4-113 Sec. 4.10.8). Here, because the IEEE Floating Compare instruction includes a target register (Fc) which stores the outputted results of a comparison, it is inherent that a path exist between the logic which performs the comparison and the target register specified by the instruction so that the specified target register can be correctly located and written into.

13. Regarding claim 3, the *Alpha Architecture Handbook* has taught the processing core that executes the compare instruction as set forth in claim 1, wherein decode logic selects the first and second input operand registers from a plurality of general purpose registers (see p.4-113 Sec.

4.10.8). Here, because the IEEE Floating Compare instruction includes two operand registers (Fa and Fb), it is inherent that decoding logic be present in any processor which implements the Alpha instruction set architecture so that the specified operand registers can be correctly located and read from.

14. Regarding claim 5, the *Alpha Architecture Handbook* has taught the processing core that executes the compare instruction as set forth in claim 1, further comprising:

a. A first load path between the first input operand register and comparison logic (see p.4-113 Sec. 4.10.8),

b. A second load path between the second input operand register and comparison logic (see p.4-113 Sec. 4.10.8). Here, because the IEEE Floating Compare instruction includes two operand registers (Fa and Fb) which contain the operands for the comparison, it is inherent that a path exist between the logic which performs the comparison and each of the operand registers specified by the instruction so that the specified operands can be correctly located and read from, allowing the comparison to be made correctly.

15. Regarding claim 6, the *Alpha Architecture Handbook* has taught the processing core that executes the compare instruction as set forth in claim 1, wherein the output operator register stores a value indicating a relationship between the first and second input operator registers which is at least one of greater than, less than, equal to and not a number (see p.4-113, Sec. 4.10.8, Description).

16. Regarding claim 7, the *Alpha Architecture Handbook* has taught the processing core that executes the compare instruction as set forth in claim 6, wherein the not a number value indicates

a comparison between the first and second input operands that cannot be made (see p.4-113, Sec. 4.10.8, Description). Here, because the unordered comparison is true if one or both operands are NaN, it is inherent that the comparison cannot be made because NaN is not a number, and a comparison can inherently not be made between a number and something other than a number.

17. Regarding claim 8, the *Alpha Architecture Handbook* has taught the processing core that executes the compare instruction as set forth in claim 6, wherein the value is an integer (see p.4-113, Sec. 4.10.8, Description). Here, when the comparison result is false, a value of zero is written into the target register, with zero being defined as an integer.

18. Regarding claim 9, the *Alpha Architecture Handbook* has taught the processing core that executes the compare instruction as set forth in claim 1, wherein:

a. The first input operand register is a double precision floating point data type,

b. The second input operand register is a single precision floating point data type,

c. The output operand register is a double precision floating point data type (see p.3-2 Sec. 3.1.3, p.4-62 Sec. 4.7, and p.4-113 Sec.4.10.8).

19. Here, the *Alpha Architecture Handbook* has taught the IEEE Floating Compare instruction using the T_Floating data format (see p.4-113 Sec.4.10.8), which is defined as the IEEE Double Precision floating point standard (see p.4-62 Sec 4.7). If one of the registers in the instruction, whether it is an input operand register or the output operand register, is single precision, the data is stored as a single precision value (see p.4-62 Sec. 4.7) but operated on correctly by instructions that use double precision values, such as the Floating Compare instruction (see p.3-2 Sec. 3.1.3).

20. Regarding claim 11, the *Alpha Architecture Handbook* has taught the processing core that executes the compare instruction as set forth in claim 1, wherein the register file comprises special purpose registers which cannot store an output operand (see p.3-2 Sec. 3.1.3). Here, the registers F31 is a special purpose register in the floating-point register file which only supplies and stores the value of zero regardless of what is written to it.

21. Regarding claim 12, the *Alpha Architecture Handbook* has taught a method for performing a compare operation, the method comprising steps of:

a. Decoding a compare instruction (see p.4-113 Sec. 4.10.8). Here, because the IEEE Floating Compare instruction includes two operand registers (Fa and Fb) and a target register (Fc), it is inherent that decoding logic be present in any processor which implements the Alpha instruction set architecture so that the operand registers can be located and read from and the specified target register can be correctly located and written into.

b. Configuring a first and second path between a register file and a comparison logic (see p.4-113 Sec. 4.10.8). Here, because the IEEE Floating Compare instruction includes two operand registers (Fa and Fb) which contain the operands for the comparison, it is inherent that a path exist between the logic which performs the comparison and each of the operand registers specified by the instruction so that the specified operands can be correctly located and read from, allowing the comparison to be made correctly.

c. Configuring a third path between the comparison logic and the register file (see p.4-113 Sec. 4.10.8). Here, because the IEEE Floating Compare instruction

includes a target register (Fc) which stores the outputted results of a comparison, it is inherent that a path exist between the logic which performs the comparison and the target register specified by the instruction so that the specified target register can be correctly located and written into.

d. Comparing a first input operand and a second input operand to produce a result that indicates an absence of at least three mathematical relationships between the first input operand and the second input operand (see p.4-113 Sec. 4.10.8). Here, because the four different relationships offered by the IEEE Floating Compare instruction are stated to be mutually exclusive, when one of the comparisons produces a result affirming the comparison it is inherent that the other three relationships do not exist.

e. Coupling an output operand to a general-purpose register in the register file (see p.4-113 Sec. 4.10.8). Here, because the IEEE Floating Compare instruction includes a target register (Fc) which stores the outputted results of a comparison, it is inherent that a path exist between the logic which performs the comparison and the target register specified by the instruction so that the specified target register can be correctly located and written into.

22. Regarding claim 14, the *Alpha Architecture Handbook* has taught the method for performing the compare operation as set forth in claim 12, wherein the configuring steps each comprise a step of addressing a general-purpose register in the register file (see p.4-113 Sec. 4.10.8). Here, because the IEEE Floating Compare instruction includes two operand registers (Fa and Fb) and a target register (Fc), it is inherent that decoding logic be present in any

processor which implements the Alpha instruction set architecture so that the operand registers can be located and read from and the specified target register can be correctly located and written into. Furthermore, because a register can't be read from or written to without knowing its address, inherent to the locating of the operand registers contained within the IEEE Floating Compare instruction is the addressing of those registers within the register file so they can be read out and/or written to.

23. Regarding claim 16, the *Alpha Architecture Handbook* has taught the method for performing the compare operation as set forth in claim 12, wherein the comparing step comprises a step of converting a data type of at least one of the first and second input operands (see p.4-62 Sec. 4.7). Here, it is taught that data conversion takes place between single- and double-precision floating-point instructions. The *Alpha Architecture Handbook* has taught the IEEE Floating Compare instruction using the T_Floating data format (see p.4-113 Sec.4.10.8), which is defined as the IEEE Double Precision floating point standard (see p.4-62 Sec 4.7). If one of the registers in the instruction, whether it is an input operand register or the output operand register, is single precision, the data is stored as a single precision value (see p.4-62 Sec. 4.7) but operated on correctly by instructions that use double precision values, such as the Floating Compare instruction (see p.3-2 Sec. 3.1.3).

24. Regarding claim 17, the *Alpha Architecture Handbook* has taught a method for executing a compare instruction in a processor, the method comprising steps of:

a. Issuing the compare instruction. While not taught explicitly, it is inherent in the operation of a processor which implements an instruction set that in order to execute an instruction it must be issued to the execution unit(s).

b. Comparing a first input operand and a second input operand to determine at least two mathematical relationships between the first and second input operands (see p.4-113 Sec. 4.10.8, CMPTLE instruction),

c. Determining an output operand indicative of the mathematical relationships (see p.4-113 Sec. 4.10.8, Description),

d. Storing the output operand in a general-purpose register of a register file (see p.4-113 Sec. 4.10.8, Description).

25. Regarding claim 20, the *Alpha Architecture Handbook* has taught the method for executing the compare instruction in the processor as set forth in claim 17, wherein the general-purpose register is used to store operators from other types of instructions (see p.3-2 Sec.3.1.3). Here, the floating-point registers are general-purpose registers that are used as sources and targets for floating-point instructions, which means that the registers are inherently able to store operators from other types of instructions than the IEEE Floating Compare instruction.

***Claim Rejections - 35 USC § 103***

26. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

27. Claims 2, 4, 10, 15 and 19 are rejected under 35 U.S.C. 103(a) as being unpatentable over the *Alpha Architecture Handbook* as applied to claim 1 above, and further in view of Colwell et al., U.S. Patent No. 4,833,599.

28. Regarding claim 2, the *Alpha Architecture Handbook* has taught the processing core that executes the compare instruction as set forth in claim 1, but has not explicitly taught wherein a very long instruction word includes a plurality of compare instructions.

29. However, Colwell has taught a processor that executes multiple conditional branch instructions that reside in a VLIW instruction in parallel to improve execution speed and reduce the delay associated with branch mis-predictions (see Col.1 line 40 – Col.2 line 18 and Col.3 lines 3-34). One of ordinary skill in the art would have recognized that it is desirable and a goal of microprocessor design to improve the speed and throughput of a microprocessor. Therefore, one of ordinary skill in the art would have found it obvious to modify the processor of the *Alpha Architecture Handbook* to process a plurality of conditional branch instructions in a VLIW instruction in parallel to improve the execution speed of the processor.

30. Regarding claim 4, the *Alpha Architecture Handbook* has taught the processing core that executes the compare instruction as set forth in claim 1, but has not explicitly taught wherein the processing core issues a plurality of compare instructions at one time.

31. However, Colwell has taught a processor that executes multiple conditional branch instructions that reside in a VLIW instruction in parallel to improve execution speed and reduce the delay associated with branch mis-predictions (see Col.1 line 40 – Col.2 line 18 and Col.3 lines 3-34). It is inherent to the parallel execution of multiple instructions then is the parallel issue of multiple instructions. One of ordinary skill in the art would have recognized that it is desirable and a goal of microprocessor design to improve the speed and throughput of a microprocessor, and one method of doing so is to increase the instruction-level parallelism by issuing and subsequently executing multiple instructions in parallel (see Col.1 lines 12-26).

Therefore, one of ordinary skill in the art would have found it obvious to modify the processor of the *Alpha Architecture Handbook* to issue a plurality of compare instructions at one time so they can be executed in parallel and thus increase the throughput and execution speed of the processor.

32. Regarding claim 10, the *Alpha Architecture Handbook* has taught the processing core that executes the compare instruction as set forth in claim 1, but has not explicitly taught it further comprising a plurality of processing paths that are coupled to the register file.

33. However, Colwell has taught a processor with multiple processor clusters, each containing multiple processing paths (integer and floating point processors) with each processing path connecting to a register file (see Col.5 lines 47-55), so that multiple instructions can be executed in parallel each cycle, providing an increase in processor performance (see Col.1 lines 7-26). One of ordinary skill in the art would have recognized that it is desirable and a goal of microprocessor design to improve the speed and throughput of a microprocessor. Therefore, one of ordinary skill in the art would have found it obvious to modify the processor of the *Alpha Architecture Handbook* to process a plurality of conditional branch instructions in a VLIW instruction in parallel to improve the execution speed of the processor.

34. Regarding claim 15, the *Alpha Architecture Handbook* has taught the method for performing the compare operation as set forth in claim 12, but has not explicitly taught wherein a very long instruction word comprises the compare operation.

35. However, Colwell has taught a processor that executes multiple conditional branch instructions that reside in a VLIW instruction in parallel to improve execution speed and reduce the delay associated with branch mis-predictions (see Col.1 line 40 – Col.2 line 18 and Col.3

lines 3-34). One of ordinary skill in the art would have recognized that it is desirable and a goal of microprocessor design to improve the speed and throughput of a microprocessor. Therefore, one of ordinary skill in the art would have found it obvious to modify the processor of the *Alpha Architecture Handbook* to process a plurality of conditional branch instructions in a VLIW instruction in parallel to improve the execution speed of the processor.

36. Regarding claim 19, the *Alpha Architecture Handbook* has taught the method for executing the compare instruction in the processor as set forth in claim 17, but has not explicitly taught wherein the compare instruction is a very long instruction word which comprises a plurality of compare instructions which are processed in parallel down separate processing paths.

37. However, Colwell has taught a processor that executes multiple conditional branch instructions that reside in a VLIW instruction in parallel to improve execution speed and reduce the delay associated with branch mis-predictions (see Col.1 line 40 – Col.2 line 18 and Col.3 lines 3-34). One of ordinary skill in the art would have recognized that it is desirable and a goal of microprocessor design to improve the speed and throughput of a microprocessor. Therefore, one of ordinary skill in the art would have found it obvious to modify the processor of the *Alpha Architecture Handbook* to process a plurality of conditional branch instructions in a VLIW instruction in parallel to improve the execution speed of the processor.

38. Claim 13 is rejected under 35 U.S.C. 103(a) as being unpatentable over the *Alpha Architecture Handbook* as applied to claim 12 above, and further in view of Patterson and Hennessy, *Computer Organization and Design*.

39. Regarding claim 13, the *Alpha Architecture Handbook* has taught the method for performing the compare operation as set forth in claim 12, the method further comprising a step of enabling the comparison logic in an arithmetic logic unit.

40. However, Patterson and Hennessy have taught that logical operations, such as comparisons, are conventionally performed within an arithmetic logic unit because they contain the necessary hardware building blocks to perform comparisons, such as AND gates for equality comparisons (see p.230-231). One of ordinary skill in the art would have recognized the desire of a microprocessor designer to reuse hardware in order to minimize the space taken up by dedicated hardware on the chip. Therefore, one of ordinary skill in the art would have found it obvious to modify the processor of the *Alpha Architecture Handbook* to include its comparison logic in an arithmetic logic unit, so as to adhere to convention and reuse hardware which already exists, thus minimizing the spaced needed for dedicated hardware on the chip.

41. Claim 18 is rejected under 35 U.S.C. 103(a) as being unpatentable over The *Alpha Architecture Handbook.*

42. Regarding claim 18, the *Alpha Architecture Handbook* has taught the method for executing the compare instruction in the processor as set forth in claim 17, but has not explicitly taught wherein the comparing step has the ability of:

a. Determining if the first input operand is less than the second input operand,

b. Determining if the first input operand is greater than the second input operand,

c. Determining if the first input operand is equal to the second input operand,

d. Determining if there is no valid relationship between the first input operand and the second imputer operand.

43. The *Alpha Architecture Handbook* has taught the above comparisons as part of four different instructions (see p.4-113 Sec. 4.10.8), but has not explicitly taught a single step with the ability to perform the four comparisons simultaneously. However, the comparison logic required to perform the four comparisons as described inherently exists (see paragraph 11 above). One of ordinary skill in the art would have recognized that executing the comparisons of four instructions as a single instruction in a single clock increases the speed and throughput of a microprocessor. Therefore, one of ordinary skill in the art would have found it obvious to perform the four comparisons simultaneously when processing one instruction instead of separately for each of four instructions in order to increase the speed and throughput of the processor.

***Conclusion***

44. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. Applicant is reminded that in amending in response to a rejection of claims, the patentable novelty must be clearly shown in view of the state of the art disclosed by the references cited and the objections made. Applicant must also show how the amendments avoid such references and objections. See 37 CFR § 1.111(c).

45. Morrison et al., U.S. Patent No. 5,517,628, has taught a condition code register file which can be addressed from within an instruction, allowing a user selectable register to be used to store the condition codes for the current processor context.

46. Chayut, U.S. Patent No. 6,574,728, has taught an arithmetic logic unit which performs compare operations communicating with a condition code stack which stores the results of the comparison operations.

47. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Barry J. O'Brien whose telephone number is (703) 305-5864. The examiner can normally be reached on Mon.-Fri. 7am-4:30pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (703) 305-9712. The fax phone number for the organization where this application or proceeding is assigned is (703) 872-9306.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

Barry J. O'Brien
Examiner
Art Unit 2183

BJO
1/23/2004

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100